AF/# 2700
#7
CG 9/27/02
29-07

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Lais et al.
Serial No.:	09/507,261
Filed:	February 18, 2000
Group Art Unit:	2187
Confirmation No.:	9010
For:	Directory Free Multinode Computer System

CERTIFICATE OF MAILING UNDER 37 C.F.R. 1.8(a)

I hereby certify that the enclosed or attached correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231, on September 20, 2002.

Pryor A. Garnett, Reg. No. 32,136

Box AF
Commissioner for Patents
Washington, D.C. 20231

APPEAL BRIEF TRANSMITTAL

Transmitted herewith for filing is an Appeal Brief in the above-identified Application. Three copies of the Brief, including Appendices (if any), are enclosed as required under 37 C.F.R. 1.192.

Deposit Account Authorization

☒ Please charge Deposit Account No. 501336 in the amount of $ 320.00, the fee required under 37 C.F.R. 1.17(c) for filing this Appeal Brief. A duplicate copy of this sheet is enclosed.

☒ The Commissioner is hereby authorized to charge payment of the following fees associated with this communication or credit any overpayment to Deposit Account 501336. A duplicate copy of this sheet is enclosed.

- ☒ Any additional filing fees required under 37 C.F.R. §1.16.
- ☒ Any patent application processing fees under 37 C.F.R. §1.17.

Request for Extension of Time

☒ Applicants respectfully request a **two (2)** month extension of time to the period for filing the Appeal Brief. Please charge Deposit Account 501336 in the amount of $ 400.00. A duplicate copy of this sheet is enclosed.

Respectfully submitted,

By:

Pryor A. Garnett
Attorney for Applicants
Registration No. 32,136

IBM Corporation
IPLaw Dept., EDO2-06
15450 SW Koll Parkway
Beaverton, OR 97006-6063

(503) 578-5020 voice
(503) 578-5040 fax

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Lais et al.

Serial No.: 09/507,261

Filed: February 18, 2000

Group Art Unit: 2187

Confirmation No.: 9010

For: Directory Free Multinode Computer System

Commissioner for Patents
Washington, D.C. 20231

APPEAL BRIEF
IN SUPPORT OF APPEAL FROM THE PRIMARY EXAMINER TO THE BOARD OF APPEALS

This is an appeal of a Final Action rejecting claims 1-20 of Application Serial Number 09/507,261 filed February 18, 2000. This brief is being submitted pursuant to 37 C.F.R. 1.192. A Notice of Appeal was filed on June 18, 2002. A two-month extension of time for submitting this Brief is being filed concurrently herewith.

1. Real Party in Interest

International Business Machines Corporation is the real party in interest.

2. Related Appeals and Interferences

There are no related appeals or interferences pending with this application.

3. Status of the Claims

Appellants appeal from the rejection in the June 4, 2002 Final Action of claims 1-20. The claims on appeal are set forth in Appendix A.

4. Status of Amendments

No amendments were filed subsequent to the Final Action of June 4, 2002.

5. Summary of the Invention

As explained in the Specification in the last two paragraphs of page 3, the goal of the invention is "to simplify existing architectures and the protocol used for inter-node communication," "to reduce costs in a multinode computer system by reducing the complexity of the protocol and/or hardware used to communicate between the nodes," and "to ensure that the multinode computer system maintains forward progress of requests for lines."

As described in the specification on page 4, according to the invention:

> "In one aspect of a multinode computer system, the local cache directory is eliminated. Prior protocols used in multinode systems, such as the SCI protocol, require a local cache directory to be present. By eliminating the directory, communication between nodes is simplified and hardware costs are reduced. In two-node systems according to the invention, lines are implied to be either on the local node, shared between the nodes, or cached at the remote node. Thus, if a local node does not have a valid copy, it is implied that the other node in the system has the valid copy.
>
> "Another aspect of the described system is a "silent rollout." In previous distributed memory multiprocessor systems, when a remote node has capacity limitations, it must overwrite (i.e., rollout) a cache line and report to the home node that the rollout occurred. However, the present system allows the remote node to rollout a cache line without reporting to the home node that the rollout occurred. Such a silent rollout can create timing problems because the home node still believes the remote node has a shared copy of the cache line.

> "Another aspect of the described system concerns using a state machine to solve timing problems caused by silent rollouts in an unordered network. After a silent rollout, if the remote node attempts to obtain a shared copy of the cache line at about the same time that the home node attempts to write the line, the home node may send the remote node an invalidate message instead of the requested line. A remote node cannot discern the difference between an invalidation for a previously rolled out line or for the data response still pending. This scenario can occur repeatedly, preventing the remote node from achieving forward progress. To ensure forward progress, if the remote node requests a line and receives an invalidate message, the state machine ensures that a request for an exclusive copy of the line is issued by the remote node. By requesting an exclusive copy, the remote node is guaranteed to obtain the desired line and forward progress is achieved. Thus, even when the remote node only desires a shared copy of the line, it may request an exclusive copy to ensure forward progress. Such timing problems may occur because the system is an unordered network, meaning that responses may be received out of order with respect to their corresponding requests."

6. Issue

The Examiner has rejected claims 1-20 under 35 U.S.C. § 102 as being anticipated by U.S. Patent 5,802,578 (hereafter *Lovett*). The sole issue on appeal is whether the Examiner is correct in asserting that claims 1-20 are anticipated by the *Lovett* reference.

7. Grouping of Claims

Each of the independent claims is independently patentable for the reasons explained below. Their respective dependent claims contain limitations that further distinguish them from the prior art, and they are therefore separately patentable as well. However for purposes of this Appeal, Appellants group the claims into two groups:

A. Claims 1-18 stand or fall together.

B. Claims 19 and 20 separately stand or fall together .

8. Argument

The Examiner rejected claims 1-20 under 35 U.S.C. § 102 as being anticipated by *Lovett.* Appellants respectfully disagree with the Examiner's position and submit that claims 1-20 are not anticipated by *Lovett.*

A. Claims 1-18

The Claimed Invention

Each of independent claims 1, 10, 14 and 18 recites as an element of the claimed invention, in slightly varying language, responding to a received invalidate request with a request for an exclusive copy. Claim 1 requires, "in response to the request to invalidate the cache line, requesting an exclusive copy of the cache line." Claim 10 requires, "receiving, on the remote node, a request to invalidate the cache line before the requested copy of the cache line is received; and issuing a request for an exclusive copy of the cache line so that the remote node can obtain control of the cache line." Claim 14 requires that "when an invalidate request is received directly after a request is made for a copy of the memory block of interest, the state machine automatically sends a request for an exclusive copy of the memory block." Claim 18 requires a state machine which "in response to the invalidate message issues a request for an exclusive copy of the data."

These independent claims thus all require, in pertinent part, first receiving an invalidate request for that line, and then in response requesting an exclusive copy of that line. In other words, these independent claims require responding to a received invalidate message with a request for an exclusive copy.

The Rejection

The Examiner rejected these independent claims in the application on identical grounds, both in the first Office Action dated December 21, 1001, and in the Final Action identified above. In pertinent part, those rejections stated that the claim limitation, "in response to the request to invalidate the cache line requesting an exclusive copy of the cache line" is taught by *Lovett*'s "updating a cache line which is in a state indicating it is the 'only cached copy' either consistent with memory or inconsistent with memory (e.g. see col. 5, Table 1)." In the Final Action's "Response to Applicants' remarks" the Examiner explained,

> "(T)he invalidation must take place before (the) remote node can have exclusive ownership of the cache line. No two nodes can have the ability to change the content of the cache line due to coherence constraints within the system. Additionally, the changing of the status of cache lines within the memory subsystems is not instantaneous but the status changes occur while other activities are taking place. An exclusive state can only be granted to a requesting node if the other copies of the cache line are put in a status which does not allow changes to the cache line by any other node. This would require invalidation of the other copies of the cache lines."

The *Lovett* Reference

The Examiner is correct in asserting that the cache states CS_ONLY_FRESH and CS_ONLY_DIRTY of *Lovett*'s Table 1 and Fig. 5 refer to states in which an exclusive copy of a cache line is stored.

However nothing in *Lovett* teaches responding to a received invalidate request with a request for an exclusive copy. At col. 8 lines 32-25, *Lovett* describes sending invalidate requests to other nodes, but not receiving and processing such invalidate requests by those other nodes. The subsequent portion of *Lovett* describes how, in a node storing an invalidated line, a processor requesting that non-invalidated line from local memory can obtain the line's valid content (e.g. col. 8 lines 45-67). *Lovett* is not concerned with, and does not teach, how a node should respond to a received invalidate request.

Lovett Does Not Anticipate Claims 1-18

MPEP § 706.02 sets forth the standard for a proper 35 U.S.C. § 102 rejection: "the reference must teach every aspect of the claimed invention either explicitly or impliedly. Any feature not directly taught must be inherently present." A feature is inherent only if it is "matter (that) is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill." M.P.E.P. § 2131.01 III.

As explained above, Lovett does not explicitly teach responding to a received invalidate request with a request for an exclusive copy. *Lovett* describes sending invalidate requests to other nodes, but not receiving and processing such invalidate requests by those other nodes, much less responding to them by sending a request for an exclusive copy. *Lovett* is not concerned with, and does not teach, how nodes should respond to received invalidate requests.

Nor is that claim element inherent in *Lovett*, contrary to the Examiner's apparent assertion in the "Response to Applicant's remarks" in the Final Action (quoted above). Whether or not *Lovett* teaches that the "granting of an exclusive state on a cache line requires invalidating other copies of that cache line," it is not inherent therein that a received invalidate response be responded to with a request for an exclusive copy. At most, that teaching would argue that it is inherent in granting exclusivity to first send an invalidate request for that line. And interpreting *Lovett*'s teaching at col. 8 lines 32-35 regarding invalidate requests most generously, it is inherent therein only that the invalidate requests be responded to — most probably with an acknowledgment of invalidation; perhaps with another request for a shared copy, resulting in the prior art looping behavior described in the Summary of the Invention. Nothing in *Lovett* inherently requires that the response from one of the remote nodes be a request for an exclusive copy, as required by the rejected independent claims.

Because *Lovett* neither explicitly nor inherently teaches the limitation (in slightly varying language) in each of independent claims 1, 10, 14 and 18 of responding to a received invalidate request with a request for an exclusive copy, Applicants respectfully request reversal of the Examiner's rejection of claims 1-18 as being anticipated by *Lovett*.

B. Claims 19 and 20

The Claimed Invention

Independent claim 19 recites in its first two claim elements, nodes which do not have cache directories:

> "providing a first node ... wherein the first node does not have a cache directory;
>
> "providing a second node ... wherein the second node does not have a cache directory."

The Rejection

The Examiner rejected claim 19 as anticipated by *Lovett* for the same reasons given for claims 1, 10, 14 and 18. However none of those reasons addressed the presence of a cache directory. The basis for the rejection of claims 19 and 20 is therefore unknown.

The *Lovett* Reference

Lovett teaches using a chained directory scheme having cache tags **36** to determine which node has the valid data for a block that is not in local memory (col. 4 lines 58 - col. 5 line 2):

> "To maintain coherence among the remote caches in the nodes of computer system 10, the system uses a chained directory scheme defined in tthe SCI protocol. The chained directory is implemented as a doubly linked sharing list that keeps track of which caches share the same data. With this sharing list, remote caches can be notified when the data they share has been changed and thus is no longer valid. ... The remote cache tag is linked into the chained directory to indicate that the node's remote cache 44 contains a copy of the shared data."

Lovett's uses a directory "that keeps track of which caches share the same data", storing in that directory the cache tags 36 which indicate "whether the data is valid and, if not, on which node the valid data can be found" (col. 7 lines 16-17). Thus, *Lovett* teaches a system with first and second nodes which do have cache directories, and using cache tags 36 in those directories to determine "on which node the valid data can be found." Nothing in *Lovett* teaches or suggests skipping over the directory when determining the location of a memory block, and the Examiner has not presented any argument that such a teaching is inherent in the *Lovett* system.

Lovett Does Not Anticipate Claims 19 and 20

Claim 19 requires nodes which do not have cache directories. *Lovett* teaches only nodes with cache directories, and neither explicitly nor inherently teaches obtaining memory blocks in nodes not having cache directories. The *Lovett* reference therefore does not teach every aspect of the claimed invention, and cannot anticipate claims 19 and 20. Applicants therefore respectfully request reversal of the Examiner's rejection of claims 19 and 20 as being anticipated by *Lovett*.

9. Summary

For the foregoing reasons, Applicants submit that the Examiner's final rejection of claims 1-20 was erroneous, and respectfully request reversal of that rejection.

Respectfully submitted,

By: ____________________

Pryor A. Garnett
Attorney for Applicants
Registration No. 32,136

IBM Corporation
IPLaw Dept. – EDO2-06
15450 SW Koll Parkway
Beaverton, Oregon 97006-6063

Telephone: (503) 578-5020
Fax No.: (503) 578-5040

Appendix A

Claims

1. In a multinode computer system with distributed shared memory, a method of ensuring that a remote node receives a copy of a cache line stored on a home node, comprising:

from a remote node, requesting a shared copy of a cache line that is stored on a home node;

while waiting for the requested cache line, receiving on the remote node a request to invalidate the cache line; and

in response to the request to invalidate the cache line, requesting an exclusive copy of the cache line.

2. The method of claim 1, further including storing a shared copy of the cache line on the remote node and rolling out the cache line on the remote node prior to the request for a shared copy of the cache line, without informing the home node of the rollout.

3. The method of claim 1, wherein the remote and home nodes do not use a directory to track the locations of cache lines.

4. The method of claim 1, further including issuing a request for the cache line from a processor on the home node at approximately the same time the request for the shared copy of cache line is received from the remote node.

5. The method of claim 1, further including discarding a response to the request for the shared copy of cache line after receiving the invalidate request.

6. The method of claim 1, further including passing data between the nodes using a system interconnect that includes a dualport RAM controlled by at least one state machine.

Appendix A

Claims

7. The method of claim 1, further including providing a state machine in the remote node that upon requesting the cache line, remains in a first pending state;

if while in the first pending state, the cache line is received, storing the cache line into a cache on the remote node and transitioning to a dirty or fresh state;

if while in the first pending state, a request to invalidate is received, transitioning to a second pending state; and

while in the second pending state, upon receiving the cache line, discarding the cache line and issuing the request for an exclusive copy of the cache line.

8. The method of claim 1 wherein the multinode computer system includes multiple processors on each node and the multinode computer system is an unordered network.

9. The method of claim 1 wherein the multinode computer system includes at most two nodes.

10. In a multinode computer system with distributed shared memory, a method of ensuring that a remote node receives a copy of a cache line stored on a home node, comprising:

storing a copy of a cache line on the remote node, wherein the cache line is a shared copy of the cache line that is also stored on the home node;

overwriting the cache line on the remote node without informing the home node that the cache line is no longer stored on the remote node;

after overwriting the cache line, requesting a new copy of the cache line be delivered to the remote node again;

receiving, on the remote node, a request to invalidate the cache line before the requested copy of the cache line is received; and issuing a request for an exclusive copy of the cache line so that the remote node can obtain control of the cache line.

11. The method of claim 10 further including:

receiving a response on the remote node for the requested cache line after receiving the invalidate request; and

discarding the response on the remote node for the requested cache line;

12. The method of claim 10 further including:

receiving a processor request on the home node for control of the cache line;

in response to the processor request, checking local tags to determine if another node has a shared copy of the cache line; and

upon determining that another node has a shared copy, sending the invalidate request to the remote node because the home node was not notified that the remote node has overwritten its copy of the cache line.

13. The method of claim 10 further including receiving on the remote node an exclusive copy of the cache line.

14. A multinode computer system with distributed shared memory that ensures a remote node receives a copy of data stored on a home node, comprising:

a first node having multiple processors, a local memory, and a remote cache, wherein the first node is a home node that stores a memory block of interest;

a second node having multiple processors, a local memory, and a remote cache, wherein the second node is a remote node with regard to the memory block of interest;

a system interconnect coupling the first node to the second node; and a state machine located on the second node, the state machine moving between different states in response to requests made from the second node to the first node and responses to the requests;

the state machine having states such that when an invalidate request is received directly after a request is made for a copy of the memory block of interest, the state machine automatically sends a request for an exclusive copy of the memory block of interest.

15. The multinode computer system of claim 14 wherein the first and second nodes include snoopy cache protocol engines.

16. The multinode computer system of claim 14 wherein the first and second nodes do not use a directory or list of any kind to track the location of cache lines that are stored in cache on another node.

17. The multinode computer system of claim 14 wherein the remote node performs silent rollouts of data.

18. A multinode computer system with multiple processors on the nodes that ensures a remote node obtains control of data, comprising a state machine that monitors whether an invalidate message for data is received after issuing a request for a shared copy of the data and in response to the invalidate message issues a request for an exclusive copy of the data.

19. In a multinode computer system with distributed shared memory, a method of ensuring that a remote node receives data efficiently, comprising:

providing a first node having multiple processors, memory and a remote cache, the first node being a home node with respect to a memory block of interest and wherein the first node does not have a cache directory;

Appendix A

Claims

providing a second node having multiple processors, memory and a remote cache, the second node being a remote node with respect to the memory block of interest and wherein the second node does not have a cache directory;

receiving a request on the second node from a processor on the second node for the memory block of interest;

checking remote cache on the second node to determine whether the memory block of interest is stored in the remote cache;

if the memory block of interest is not stored in the remote cache, automatically assuming that the memory block of interest is stored on the first node since the second node cannot check a cache directory to determine the location of the memory block of interest; and

sending a request to the first node for the memory block of interest.

20. The method of claim 19 further including:

receiving the memory block of interest on the second node; and

rolling out the memory block of interest on the second node without informing the first node of the rollout.

End of Appendix A